View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Universiteit Twente Repository

A Framework for the Evaluation of Semantics-based Service Composition Approaches

Eduardo Silva, Luís Ferreira Pires, Marten van Sinderen
Centre for Telematics and Information Technology
University of Twente, The Netherlands
P.O. Box 217, 7500 AE Enschede
Email: {e.m.g.silva, l.ferreirapires, m.j.vansinderen}@cs.utwente.nl

***Abstract*—The benefits of service composition are being largely acknowledged in the literature nowadays. However, as the amount of available services increases, it becomes difficult to manage, discover, select and compose them, so that automation is required in these processes. This can be achieved by using semantic information represented in ontologies. Currently there are many different approaches that support semantics-based service composition. However, still little effort has been spent on creating a common methodology to evaluate and compare such approaches. In this paper we present our initial ideas to create an evaluation framework for semantics-based service composition approaches. We use a collection of existing services, and define a set of evaluation metrics, confusion matrix-based and time-based. Furthermore, we present how composition evaluation scenarios are generated from the collection of services and specify the strategy to be used in the evaluation process. We demonstrate the proposed framework through an example. Currently there are mechanisms and initiatives to address the evaluation of the semantics-based service discovery and matchmaking approaches. However, still few efforts have been spent on the creation of comprehensive evaluation mechanisms for semantics-based service composition approaches.**

***Keywords*-Semantic Services; Service Composition; Evaluation and Benchmarking.**

I. Introduction

Service-Oriented Computing (SOC) [1] is being adopted by industry as an approach to deliver network-based application software to clients and to support business-to-business collaborations. It offers message-oriented and technology-independent mechanisms, which facilitates interoperability between different and heterogeneous systems. The adoption of the Service-Oriented Architecture (SOA) [2] principles fosters the availability of large sets of services in different domains. Consequently, service composition emerges as a new approach for distributed programming, where new application services are created out of available component services. Service composition provides a higher level of abstraction in the development process, leading to a shorter development time, and to the optimisation of resources usage (technological and human) through the re-use of existing services. However, as the number and the complexity of services increase, it becomes difficult to manually manage, discover, select and compose them. To tackle this complexity approaches and techniques are required to automate the phases of the service composition life-cycle. One possible approach is the use of semantic information to describe services by defining this information ontologies. From now on we refer to semantically described services as *semantic services*. Semantic services are described on a machine readable and understandable formalism, which allows one to create tools to automate some, or all, of the phases of the service composition life-cycle, i.e., management, discovery, selection and composition of semantic services.

There are several semantics-based service composition approaches [3] [4] [5]. However, little effort has been spent to define a common evaluation framework for semantics-based service composition approaches. The proposed approaches are normally evaluated in an *ad-hoc manner* by the authors of the composition approaches. In this paper we present our initial ideas for the definition of a framework for evaluation and comparison of semantics-based service composition approaches. We assume that the approaches perform automated service discovery, selection and composition, based on a service request, and retrieve all the resulting matching compositions. We specially focus on the composition phase of the service composition life-cycle, since some work has already been done on evaluation methodologies for the discovery and selection phases [6] [7]. However, since the composition phase is proceeded by discovery and selection of services, the evaluation of these phases is also implicitly captured in the proposed evaluation framework. Our evaluation framework uses a set of publicly available services and defines evaluation metrics and an evaluation strategy. To the best of our knowledge this is one of the first attempts to define a framework for the evaluation of semantics-based service composition approaches.

This paper is further organised as follows: Section II introduces and characterises the semantics-based service composition evaluation problem; Section III discusses issues and requirements for the design of an evaluation framework for semantics-based service composition approaches, referring to the state-of-the-art in the area; Section IV presents our evaluation framework; Section V gives an example of definition of evaluation scenarios, and evaluation of semantic service composition approaches according to the proposed framework; and Section VI provides our conclusions and directions for future work.

II. Generic Problem Definition

An evaluation methodology aims at determining the quality of different semantic service composition approaches. The evaluation methodology evaluates these approaches by examining the created compositions given a service request and a service collection. Furthermore, the quality of the compositions found by an approach can also be evaluated. In the following we characterise the problem of semantics-based service composition evaluation by describing the architecture of the system that supports the evaluation, the properties of the services used in the evaluation, how these services can be used for evaluation, and finally which metrics can be used in the evaluation.

A. General System Architecture

Figure 1 presents the architecture of our evaluation system, identifying the different parts of the evaluation process, as well as the interactions and stakeholders.

We identify two main stakeholders: the evaluation framework *Designer* and the *Evaluator*. The *Designer* focuses on the creation of common evaluation scenarios to be used in the evaluations. The *Evaluator* is responsible for the evaluation of service composition approaches. The evaluation is performed by using the artefacts produced by the *Designer* and the established evaluation metrics.

Figure 1 depicts the flow of activities performed to generate the necessary artefacts for the evaluation process (steps i, ii and iii). Furthermore, the evaluation process is also depicted in the figure (steps iv and v). The generation of the evaluation scenarios consists of defining a set of common service requests, created based on the set of services available in the framework, and ontologies used to semantically annotate these services. Services and ontologies may be defined by the *Designer* or may be imported from external sources. Based on the set of service request (SR) the *Designer* defines or finds a set of correct and meaningful reference service compositions (RSC) that match and fulfil the corresponding SR. After this process, SRs and their matching RSC can be made available to possible *Evaluators* together with the framework services collection and ontologies. Based on these artefacts, the *Evaluator* follows the evaluation strategy defined in the framework and uses the framework metrics to report on the quality of the approach. Evaluation strategy and evaluation metrics are specified in the framework, in order to allow different composition approaches to be compared.

B. Semantic Services Collection

The semantic services collection is the set of semantic services used in the service composition process evaluation. It consists of services annotated with semantic concepts from common ontologies. The framework ontologies may be defined by the framework *Designer* or collected from external sources. Semantic services and ontologies must be defined and made available to be used by different *Evaluators*. Thus a common setup is necessary to allow the comparison of different composition approaches. The semantic service collection has to be large enough to allow one to derive relevant evaluations. Furthermore, the semantic service collection should contain services that can be composed, otherwise it is not suitable for the evaluation of the composition approaches.

C. Evaluation Scenarios

The generation of evaluation scenarios consists of the creation of service requests (SR) and their respective matching reference service compositions (RSC), based on the considered semantic services collection. These are common for all possible *Evaluators*, which makes it possible to compare evaluation results of different composition approaches.

D. Evaluation Metrics

The evaluation metrics refer to the number and relevance of the created service compositions. Furthermore, time-based evaluations can also be considered to measure the processing time performance associated with each composition approach.

Qualitative metrics can also be used to characterise and position different approaches with respect to each other. For example, expressivity of service description language and the ontologies used by the composition approach can be compared.

III. Design Issues and Requirements

Based on our general problem definition in this section we address issues and requirements related to the design of our evaluation framework.

A. Semantic services collection

The creation of a large and representative collection of semantic services is not a trivial task. Nowadays there are few available collections with few hundred semantic services. Some work is being done to create larger semantic services collections. Generally two directions are being taken: gather existing services and semantically annotate them, or automatically generate collections of semantic services. In the first direction *real world services* are used, which are based on services that exist and have implemented and used in practice. In the second direction semantic services are artificially created, i.e., created solely for the purpose of testing and evaluation.

1) Existing semantic services: The Semantic Web Services Challenge (SWS-Challenge)[1] [6], Service Semantic Service Selection (S3)[2] Contest, and the Semantic Web Services Test Collection (SWS-TC)[3] [8] are relevant semantic services collections based on realistic services. The SWS-Challenge offers a set of services that simulate a collaboration scenario. The focus of these services is on the evaluation of the suitability of the approaches to solve a mediation problem. This services collection is limited, containing around a dozen services. In the SWS-Challenge, services are specified in detailed natural

[1] http://sws-challenge.org

[2] http://www-ags.dfki.uni-sb.de/ klusch/s3/

[3] http://projects.semwebcentral.org/projects/sws-tc/

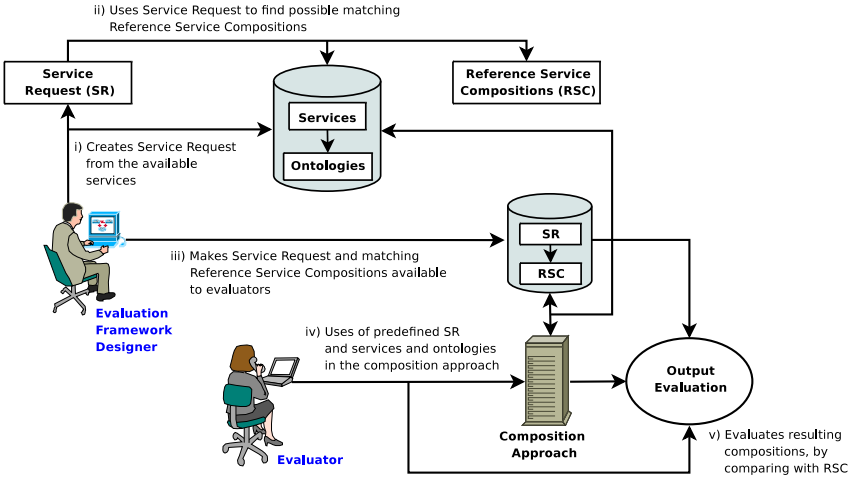

Figure 1. General System Architecture

language descriptions, and they have to be translated to the formalisms used by each participant in the challenge. The S3 contest offers a larger collection of services, containing more than a thousand services. This collection was created from existing (web) services description documents (WSDL descriptions) by extending them with semantic annotations, as references to a set of selected ontologies. This collection has services from different domains with very different properties. However, in general these services lack implementation, and therefore the collection is criticised [9] because some of the services are not realistic, poorly specified, and are described only in some semantic service description languages, at the moment OWL-S [10] (OWLS-TC [11]) and SAWSDL [12]. The SWS-TC approach provides a set of 241 OWL-S services semantically annotated with only one ontology. This collection is not as large as the S3 collection (OWLS-TC), but services are better defined, i.e., they present clearer semantics.

Another similar approach has been reported in [7], which aims at the creation of a larger collection of services with diverse parameters and properties, from different application domains. To achieve this they started a community portal (OPOSSum)[4], where people can publish their semantic services. As the collection grows, it can be used as a testing collection in the evaluation process. The semantic services collections mentioned before have already been imported to this collection.

2) Automatic Generation of semantic services: Theoretically, a way to overcome the difficulties of creating large service test collections is to automatically generate them. In [13] and [14] two approaches are proposed to automatically generate web services. These approaches allow to specify some characteristics and parameters, e.g., number of input/output parameters, etc., which are then used to guide the generation of a user defined number of services. These approaches do not support the generation of semantic services, but they could be seen as a starting point for this. In order to generate a realistic set of semantic services, existing semantic services have to be investigated and characterised to capture their properties. Based on the result of such study service collections could be generated. However, this process has some complex problems, which concern the ontology concepts used to describe the services parameters. A pure random annotation of services is not realistic, since the evaluation may not be meaningful. This yields to evaluation results that will not be representative of real situations. The semantics of real services is difficult to model. Another complex problem is to model classes of services, i.e., services that are similar or share common parameters and properties. These are possibly the reasons why there is a lack of solutions to automatically generate semantic services. From the best of out knowledge the Web Services Challenge[5] is one of the initiates that use automatic generated semantic service collections.

B. Generation of Evaluation Scenarios

The generation of service composition scenarios consists of defining common *service requests* to be used in the service composition process, based on a common *semantic services collection*. We consider two distinct ways to generate evaluation scenarios: i) introduce a set of services in the services collection, which can generate a service composition capable of fulfil a specified service request (*top-down* approach); ii) create a set of service requests and respective matching service compositions from the semantic services collection (*bottom-up* approach).

The two approaches differ in several aspects. For example, the *top-down* approach introduces services that lead to possible service compositions. In contrast, the *bottom-up* approach depends on the used service collection to allow service composition, i.e., some collections may not allow semantic service composition. The *top-down* approach has

[4]http://fusion.cs.uni-jena.de/opossum/

[5]http://ws-challenge.georgetown.edu/

some practical disadvantages, since it is not trivial to generate large collections of service manually.

An alternative solution can be found by combining these approaches, for example consider a large collection of services and introduce semantic services in the existing collection so that the resulting collection guarantees the existence of service compositions that fulfil some SR.

Another issue associated with the *bottom-up* approach is the identification of the reference service compositions for the service requests. This may be difficult to be performed manually, specially when large sets of services are considered. To overcome this, a composition approach can be used to find the reference service compositions for a given service request. We denote this composition approach as the *reference service composition* approach. We assume that the reference service composition approach is not complete and may retrieve invalid service compositions, i.e., may not find all the matching service compositions for a given service request and some retrieve compositions may not be correct. To cope with this, and ensure that only valid and correct service compositions are considered, the evaluation framework designer has to perform a manual check of the proposed service compositions, filtering invalid compositions. The resulting set of correct compositions define the set of reference service compositions for the considered service request.

C. Evaluation Metrics

Different metrics are necessary to evaluate a semantics-base service composition approach. We consider only metrics based on *confusion matrix* values and on time measures. Some approaches to evaluate semantic services discovery and matchmaking propose similar metrics [7].

1) Confusion Matrix-based: A *confusion matrix* [15] contains information about actual and classified values proposed by a "classification system". A confusion matrix is a two-by-two matrix, as shown in Table I, where columns represent actual *positive* and *negative* values, and rows represent *positive* and *negative* classifications performed by a given system.

True positives are values evaluated as true when they are *actually* true; *False Positives* are values evaluated as positive when they are *actually* false; *False Negatives* are values evaluated as negative when they are *actually* positive; and *True Negatives* are values evaluated as negative when they are *actually negative*.

In the case of service composition evaluation, we assume that service composition approaches only make *positive classifications* (P'), which correspond to the compositions found for a given service request. The *negative classifications* (N') are compositions that were not found by the service composition approach. Figure 2 shows a Venn diagram for all the possible classifications of the outcomes of a service composition approach.

		Actual Values	
		P	N
Classified Values	P'	True Positives (TP)	False Positives (FP)
	N'	False Negatives (FN)	True Negatives (TN)

Table I
CONFUSION MATRIX

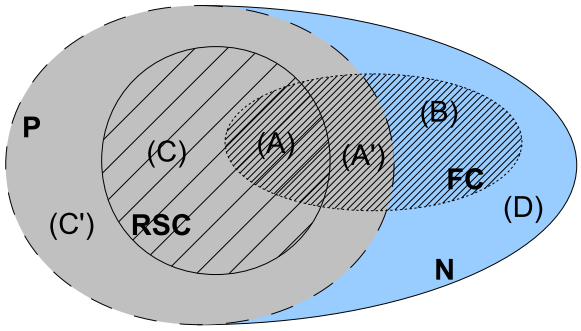

Figure 2. Venn diagram for classifications

VC (Valid Compositions) is a subset of FC (Found Compositions), which are compositions found by the composition approach being evaluated that fulfil the service request (SR) used by the evaluator. RSC are the reference service compositions for the used SR. P represents the whole set of existing valid compositions, while N represents all the other compositions. N is not known, since it represents compositions that are not selected. $RSC \subseteq P$, the *reference service compositions*, are the only known subset of valid compositions when some approach is evaluated. However, since we assume beforehand that this set may not be complete (see Section III-B) there may be a subset of valid compositions returned by the approach being evaluated that is correct but does not belong to the *reference service compositions*. This subset is defined as A' in Figure 2, where $A' = P \cap FC \setminus RC$. This subset is defined by the evaluators when the found compositions are assessed. We call this subset *valid compositions found by judgement* - $FC^P_{Judge} = A'$. The remaining valid compositions that were not found neither are part of the RSC, are unknown and are not considered in the evaluation process. They correspond to the subset $C' = P \setminus (P \cap (RC \cup FC^P_{Judge}))$. Based on these assumptions, we can define the confusion matrix for our evaluation framework in Table II.

Given this confusion matrix, we can derive several evaluation metrics, namely, *accuracy*, *precision* and *recall*, among others (see Section IV-B). These evaluations are made relative to a *reference service composition approach*. We do not assume that the reference service composition approach is complete, i.e., it may not identify all the possible service compositions that match a given service request. However, with these reference measures, different composition approaches can be evaluated and compared.

2) Time-based: Time-based performance metrics allow to compare the time performance and scalability of different approaches when the service collection size varies, among other things. These measures are extremely relevant for composition approaches that target automated service composition at runtime, since *real-time* requirements have to be met. Metrics such as the time required to perform the service composition process can be used in this

		Actual Values	
		P	N
Classified Values	P'	$TP = A \cup A'$ $TP = FC^{P}_{Judge} \cup (FC \cap RC)$	$FP = B$ $FP = FC \setminus (FC^{P}_{Judge} \cup (FC \cap RC))$
	N'	$FN = C \cup C'$, (C' is unknown) $FN \simeq RC \setminus (FC \cap RC)$	$TN = D$, D is unknown $TN = Unknown$

Table II
COMPOSITION APPROACHES CONFUSION MATRIX

form of evaluation. However, these metrics depend on the hardware, operating system, machine load, among other things, which we define as the *execution environment*. Given that they are dependent on the *execution environment* they are not suitable for *absolute value comparison*. Therefore, either another performance metric should be defined to factor out these metrics from the *execution environment*, or the Evaluators have to use these metrics simply as an indication of the performance of their service composition approaches, and not for comparison. The first alternative is difficult to realise; typically it would consist of counting the number of instructions performed on the course of the execution of the composition process, which is similar to the measure of the composition algorithm complexity order. In case these metrics are *directly* used as an indication of the time performance of an approach, the Evaluators have to indicate the describe their *execution system*, so that different results can be compared.

In order to measure *scalability*, the number of services in the collection number must be iteratively increased, so that the composition time taken in each iteration can be measured. Since composition time variation, and not the absolute vales of composition time, is measured, different composition approaches can be compared using this metric, even if they are evaluated in different *execution systems*.

IV. EVALUATION FRAMEWORK

Our evaluation framework consists of a set of evaluation scenarios, evaluation metrics and an evaluation methodology. The framework has been designed to allow different composition approaches to be evaluated under the same conditions, so that results can be compared.

A. Evaluation Scenarios

We propose the creation of evaluation scenarios based on existing collections of services, corresponding to *real world* services. We argued before that currently there are not many publicly available semantic services collections. To the best of our knowledge, the S3 contest is the initiative that provides the largest collection, consisting of more than a thousand semantic services. Therefore we planned to assume this as our initial services collection. However, from an inspection of the collection, we could observe that the services in this collection do not yield compositions, which is a necessary condition for our evaluation approach. This may be explained by the initial purpose of the S3-contest collection, which was the evaluation of semantic services discovery and matchmaking. However, we still consider this collection very relevant and a possible direction of future work can be to extend this collection with some services, in the same domains, such as that the collection can yield compositions as required for the evaluation of semantics-based service composition approaches. This can be achieved, as we referred in Section III, through an *hybrid* generation of evaluation scenarios, by using the S3-contest collection services (*bottom-up*) and introducing some other services (*top-down*) that allow the services in the collection to produce compositions.

For these reasons we consider initially the SWS-TC collection, in which we could easily define some service requests and respective reference service compositions. This collection has only one ontology (*Concepts.owl*) which contains all the semantic concepts, and their relations, used to describe the different services in the collection. In Section V we present an example of some evaluation scenarios defined based on this collection.

Figure 3 shows the overall procedure taken to generate evaluation scenarios. We assume that services, ontologies, service requests (SR) and reference service compositions (RSC) remain consistent and correct, even if they are translated to other description language or formalisms.

The construction of a service composition is based on the semantic match of a service's outputs with another service's inputs. We consider the most commonly accepted semantic match [16]: exact, plugin, subsume and fail.

B. Evaluation Metrics

1) Confusion Matrix-based: Assuming the confusion matrix defined in Section III-C, we define the following set of evaluation metrics:

$$Precision\ or\ PPV = \frac{|TP|}{|TP| + |FP|} \quad (1)$$

$$Recall\ or\ TPR = \frac{|TP|}{|TP| + |FN|} \quad (2)$$

$$FDR = \frac{|FP|}{|FP| + |TP|} \quad (3)$$

$$Acc^{TP} = \frac{|TP|}{|TP| + |FN| + |FP|} \quad (4)$$

$$SNR^{TP}_{FP} = \frac{PPV}{FDR} = \frac{|TP|}{|FP|} \quad (5)$$

Positive Classified Values or Precision, Equation 1, defines the fraction of found compositions that are correct. *True Positives Rate or Recall*, Equation 2, defines the fraction of correct compositions that were found. *False Discover Rate*, Equation 3, defines the fraction of retrieved compositions that are not correct. *Positive Accuracy*, Equation 4, defines the accuracy of the composition approach, i.e., the fraction of the correct compositions retrieved over correct, not correct and other known correct

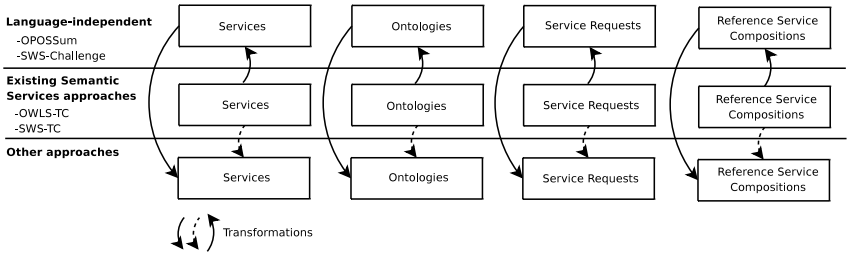

Figure 3. Generation of Evaluation Scenarios

compositions. Finally, *Signal-to-noise-Ratio*, Equation 5, presents a metric based on *signal theory*, which represents the ratio between found compositions that are correct and the found compositions that are not correct. This metric provides an overview on how *robust* against "noise" the composition approach is. This measure is important since it is not meaningful to create many service compositions from which many are not correct.

The metrics discussed above can be presented in a table, where different evaluated service composition approaches can be reported and compared. However, to have a more intuitive representation of the evaluation results, we suggest the graphical representation of the results, as shown in Figure 4. The graph represents the metrics *precision* and *recall* against the *false discovery rate* metric. If precision and recall values of an approach lie bellow the dashed line, the compositions found by a composition approach are mainly *false positives*, e.g., Approach 1 in Figure 4. On the other hand, if the precision and recall values of an approach lie above the dashed line, most of the compositions found by the composition approach are *true positives*, such as, e.g., Approach 2 in Figure 4.

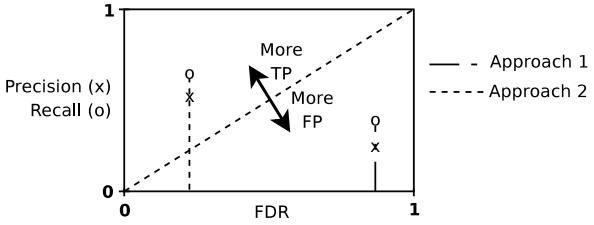

Figure 4. Metrics Graphical Representation

2) Time-based: Time-based evaluation metrics measure the time spent to perform the service composition process and how the approaches scale when the number of services considered for composition vary. We define *composition processing time* ($compProcTime$) as the time taken since a service is requested until one or more service compositions that match the service request are retrieved. When reporting on this performance metric, the users have to specify which *execution environment* was used during the evaluation process, e.g., CPU, memory and OS. This performance metric does not allow an absolute value comparison, but provides an indication of the time/processing performance of the composition approach.

The *scalability* metric is determined by the variation of the composition processing time ($compProcTime$) when the number of services considered in the composition process varies. Approaches that have a slow variation on the composition processing time have better scalability than the ones that have higher variations when the same set of services is added to the registry.

$$Scalability = \left(\frac{\partial compProcTime}{\partial \#servs} \right)^{-1} \simeq \frac{1}{N-1} \left(\sum_{i=2}^{N} \frac{compProcTime(i) - compProcTime(i-1)}{\#servs(i) - \#servs(i-1)} \right)^{-1} \quad (6)$$

Scalability, Equation 6, is inversely proportional to the change of composition time as function of the change of the number of services in the semantic services collection. We divide the services collection in N groups, which are added in each iteration of the process. Scalability is then the average of the composition time of the N iterations, i.e, $N-1$ derivatives. Since this metric is a variation and not an absolute value of the composition time, it allows different approaches to be compared, even if they have been evaluated in different *execution environments*.

C. Evaluation Methodology

Figure 5 depicts our complete methodology for the evaluation of semantics-based service composition approaches.

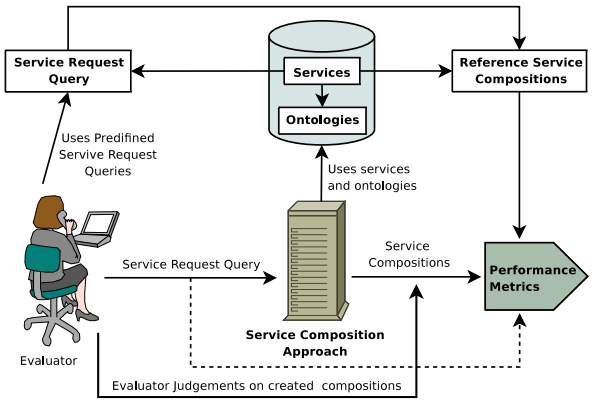

Figure 5. Evaluation Methodology

Before the evaluation takes place, the *Evaluator* has to make sure that the framework's service collections, ontologies, reference service compositions and service request are compatible with the languages/formalisms used

in his composition approach. If not, the *Evaluator* has to translate them to the formalisms of his approach, preserving the original services collection, service request and reference compositions properties. An extra effort may be necessary for these translations. However, it would be unrealistic to cover all the possible service and ontology description languages/formalisms. Similar methodology is used by some of the state-of-the-art works in the area of evaluation of semantics-based mechanisms, for example, in the SWS-Challenge.

Once the previous preconditions are satisfied, the user can make use of the predefined service requests, defined by the framework *Designer*, to start the service composition process. A service request is passed to the service composition approach, and then, depending on the approach, it is processed, allowing services to be discovered and composed. The composition time has to be monitored, so that the time-based performance metrics can be computed. For the scalability metric, the number of services in the registry must be varied, according to a defined number of iterations, and the composition time has to be measured. At the end of the service composition process, the resulting service compositions are used in conjunction with the reference service compositions to compute the proposed evaluation metrics. To compute the *confusion matrix*-based metrics, the *Evaluator* has also to evaluate the resulting compositions, so that *true positives* not defined in the *reference service compositions* can be identified.

V. Example

This section presents an example of application of the proposed framework. We first define evaluation scenarios and then we use them to perform the evaluation, according the framework methodology.

A. Semantic Services Collection

We consider the SWS-TC [8] as the semantic services collection in this example. This collection has 241 semantic services described in OWL-S, covering different domains, such as, *books information*, *hotels*, *travelling*, *weather info*, *addresses/location*, *services news*, *IT services*, etc.

We consider the services from the *books* domain to generate the evaluation scenarios. We define a service requests (SR) and find one or more matching reference service compositions (RSC). The following set of services from the SWS-TC are used initial to define SRs and RSC:

- *CheapestBookStore.owl*: Finds a bookstore that offers a book with the lowest price and returns the bookstore information and the price.
  - in: $Concepts.owl\#Book$
  - out: $\{Concepts.owl\#Store, Concepts.owl\#Price\}$
- *FindBookStore.owl*: Finds a bookstore that has the book with the input ISBN in stock.
  - in: $Concepts.owl\#ISBN$
  - out: $\{Concepts.owl\#Store, Concepts.owl\#Price\}$
- *GetBookISBN.owl*: Receives a book and finds its corresponding ISBN.
  - in: $Concepts.owl\#Book$
  - out: $Concepts.owl\#ISBN$
- *ISBNBookFinder.owl*: Receives an ISBN and returns the book information.
  - in: $Concepts.owl\#ISBN$
  - out: $Concepts.owl\#Book$
- *BookinformationFinder.owl*: Receives an ISBN and returns the book information.
  - in: $Concepts.owl\#ISBN$
  - out: $Concepts.owl\#Book$
- *BookLookup.owl*: Gets the title of a book as input and if it is in stock then returns its ISBN.
  - in: $Concepts.owl\#Text$
  - out: $Concepts.owl\#ISBN$
- *BookSearch.owl*: Returns the information about a book given its title.
  - in: $Concepts.owl\#Text$
  - out: $Concepts.owl\#Book$
- *BookPrice.owl*: Returns the price of a book if it is available in the stock.
  - in: $Concepts.owl\#Book$
  - out: $Concepts.owl\#Price$
- *AmazonBookPrice.owl*: Returns the price of a book if it is available in the stock.
  - in: $Concepts.owl\#Book$
  - out: $Concepts.owl\#Price$
- *GetbookPrice.owl*: Get the price of a book from the zwiftbooks catalogue.
  - in: $Concepts.owl\#Book$
  - out: $Concepts.owl\#Price$

More services have been defined in the *books* domain, but we limit ourselves here to the ones above to define an evaluation scenario.

B. Evaluation Scenarios

Based on services presented in the previous section, we have manually defined service composition evaluation scenarios, which consists of SR and its matching RSC. Although in this example we define the RSC manually, we argue that the RSC should be created using a reference service composition approach, given that manual identification of RSC tends to be difficult when larger semantic service collections are considered.

Service Request (SR): find a service (composition) that given a *title or description* of a book can retrieve the *Price* of the book and possibly a *Store* where the book can be bought.

Reference Service Compositions (RSC): Figure 6 provides a set of reference compositions that fulfil the defined SR. By inspecting Figure 6 we can conclude that 10 RSC were discovered that match the SR, 6 in composition A and 4 in composition B.

C. Evaluation of Composition Approach

In this section we assume the existence of two semantics-based service composition approaches: *Approach 1* and *Approach 2*. The evaluations performed here

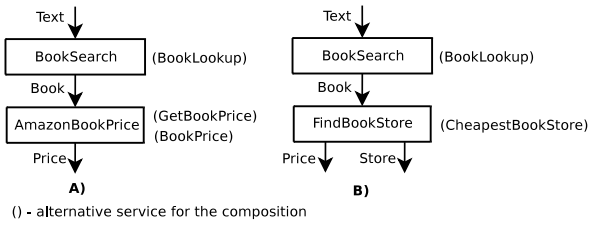

Figure 6. Reference Service Composition Set

are based on the confusion matrix-based metrics of the evaluation framework.

We assume that the *Approach 1* discovers 8 of the 10 RSC and additionally it proposes one incorrect composition (*false positive*). *Approach 1* does not create any new correct composition that does not belong to the RSC, i.e., *valid compositions found by judgement* - $FC^{P}_{Judge} = 0$. Table III presents the confusion matrix for this composition approach.

		Actual Values	
		P	N
Classified Values	P'	8	1
	N'	2	-

Table III
CONFUSION MATRIX - APPROACH 1

On the other hand *Approach 2* discovers 8 of the 10 RSC and 2 other *valid compositions found by judgement* - $FC^{P}_{Judge} = 2$. Additionally, it proposes four incorrect compositions (*false positives*). Table IV presents the confusion matrix for this composition approach.

		Actual Values	
		P	N
Classified Values	P'	10	4
	N'	2	-

Table IV
CONFUSION MATRIX - APPROACH 2

Table V shows how the compositions approaches perform in terms of the proposed evaluation framework metrics. *Approach 1* discovers less valid service compositions, but it also discovers less invalid compositions, i.e., the approach has a better accuracy (Acc^{TP}) and a higher robustness to the selection of invalid compositions (SNR^{TP}_{FP}).

Metrics	*Approach 1*	*Approach 2*
$Precision\ or\ PPV$	$8/9 \simeq 0.89$	$10/14 \simeq 0.71$
$Recall\ or\ TPR$	$8/10 = 0.8$	$10/12 \simeq 0.83$
FDR	$1/9 \simeq 0.11$	$4/14 \simeq 0.29$
Acc^{TP}	$8/11 \simeq 0.73$	$10/16 \simeq 0.63$
SNR^{TP}_{FP}	$8/1 = 8$	$10/4 = 2.5$

Table V
METRICS COMPARISON

Figure 7 gives the graphical representation of *Precision* and *Recall* against the *False Discovery Rate* (FDR), which allows us to compare how different approaches perform in the composition process. In this case, *Approach 1* has a much higher precision than *Approach 2*, and a lower FDR. This means that *Approach 1* retrieves proportionally more correct compositions than *Approach 2*, and also proportionally less incorrect compositions. In contrast, *Approach 2* is able to retrieve more valid compositions than *Approach 1* in absolute terms, higher *Recall*, however with a higher percentage of incorrect compositions.

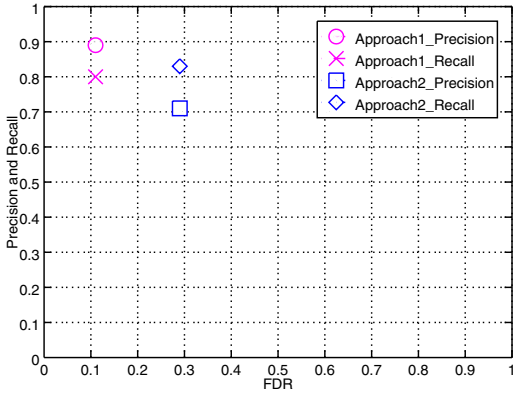

Figure 7. Reference Service Composition Set

From this discussion, and ignoring the time-based metrics, we can conclude that *Approach 1* is more appropriate to deliver *automatically* a service to an end-user, than *Approach 2*. We claim this since the probability of delivering correct compositions is higher. However, for example if we consider the case of delivering all the possible matching compositions to a service developer, so that he can customize one of the proposed compositions, *Approach 2* may also be a good candidate.

VI. CONCLUSIONS AND FUTURE WORK

This paper presents our initial ideas for an evaluation framework for semantics-based service composition approaches. Our final goal is to create a fair and practical methodology for evaluating and comparing the performance of semantics-based service composition approaches. There are already some approaches that support the evaluation of semantics-based service discovery and matchmaking. However, little effort has been spent on the problem of assessing semantic service composition approaches. To fill this gap we propose a framework based on existing collections of services. From these collections we define evaluation scenarios, by creating sets of service requests and matching reference service compositions. To evaluate the performance of existing semantic service composition approaches, we propose metrics based on a confusion matrix, and time-based metrics. The confusion matrix-based metrics allows one to assess the quality of the service compositions found by the approach under evaluation, while the time-based metrics provide an indication on time performance of the approach and its scalability with

respect to the number of available services in the semantic services collection. We have presented the application of the framework through an example, in which we defined an evaluation scenario, and then apply this scenario to two hypothetical semantics-based composition approaches. We showed that the framework's evaluation metrics allowed us to reason about different properties of the composition approaches, namely that approaches that discover more compositions may not be the most appropriate if they also retrieve a high number of incorrect compositions.

In this work we only demonstrate the feasibility of our framework using an existing collection of semantic services (SWS-TC). However, SWS-TC is a small collection and may not allow one to obtain meaningful and realistic results, namely for the time-based metrics. Therefore, in the future we intend to define a more comprehensive collection of semantic services, by considering other existing collections (S3 contest, OPOSSum). Some of these existing collection may require some new services such as they yield compositions. This is a requirement for the service collections in our framework. We will also define evaluation scenarios and use our own service composition approach [17] to create reference service compositions for each of the considered service request. Based on these results, we will apply and test the framework on some available and relevant semantics-based service composition approaches. We intend to make our framework publicly available so that interested evaluators can use it. We also plan to further extend the proposed metrics, namely to include qualitative metrics to classify the capabilities supported by different composition approaches, such as the types of composition that can be generated by the approach. These metrics may allow one to better characterise and compare the different approaches.

REFERENCES

[1] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-oriented computing: State of the art and research challenges," *Computer*, vol. 40, no. 11, pp. 38–45, 2007.

[2] T. Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall, 2005.

[3] K. Verma, K. Gomadam, A. P. Sheth, J. A. Miller, and Z. Wu, "The meteor-s approach for configuring and executing dynamic web processes," University of Georgia, Athens, Tech. Rep., June 2005. [Online]. Available: http://lsdis.cs.uga.edu/projects/meteor-s/techRep6-24-05.pdf

[4] K. Fujii and T. Suda, "Dynamic service composition using semantic information," in *International Conference on Service Oriented Computing*, New York, NY, USA, 2004, pp. 39–48.

[5] S. Kona, A. Bansal, and G. Gupta, "Automatic composition of semanticweb services," in *International Conference on Web Services*, 2007, pp. 150–158.

[6] C. J. Petrie, H. Lausen, and M. Zaremba, "Sws challenge - first year overview," in *International Conference on Enterprise Information Systems*, 2007, pp. 407–412.

[7] U. Küster and B. König-Ries, "On the empirical evaluation of semantic web service approaches: Towards common sws test collections," in *IEEE International Conference on Semantic Computing*, 2008, pp. 339–346.

[8] Y. Ganjisaffar and H. Saboohi, "Semantic web service test collection (sws-tc)," http://projects.semwebcentral.org/projects/sws-tc/, 2006.

[9] U. Küster, H. Lausen, and B. König-Ries, "Evaluation of semantic service discovery - a survey and directions for future research," in *Workshop on Emerging Web Services Technology*, November 2007.

[10] D. Martin, M. Burstein, E. Hobbs, O. Lassila, D. Mcdermott, S. Mcilraith, S. Narayanan, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara, "Owl-s: Semantic markup for web services," Tech. Rep., November 2004. [Online]. Available: http://www.w3.org/Submission/OWL-S/

[11] M. Klusch, B. Fries, M. A. Khalid, and P. Kapahnke, "Owl-s test collection (owls-tc)," http://projects.semwebcentral.org/projects/owls-tc/.

[12] J. Kopecký, T. Vitvar, C. Bournez, and J. Farrell, "Sawsdl: Semantic annotations for wsdl and xml schema," in *IEEE Internet Computing*, vol. 11, 2007, pp. 60–67.

[13] E. Cho, S. Chung, and D. Zimmerman, "Automatic web services generation," in *Hawaii International Conference on System Sciences*, 2009, pp. 1–8.

[14] S.-C. Oh, H. Kil, D. Lee, and S. R. T. Kumara, "Wsben: A web services discovery and composition benchmark," in *International Conference on Web Services*, Washington, DC, USA, 2006, pp. 239–248.

[15] R. Kohavi and F. Provost, "Glossary of terms," *Machine Learning*, vol. 30, no. 2–3, pp. 271–274, 1998.

[16] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara, "Semantic matching of web services capabilities," in *International Semantic Web Conference on The Semantic Web*, London, UK, 2002, pp. 333–347.

[17] E. Silva, J. M. López, L. F. Pires, and M. J. van Sinderen, "Defining and prototyping a life-cycle for dynamic service composition," in *International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing*, Portugal, July 2008, pp. 79–90.